

Amendments to the claims,

Listing of all claims pursuant to 37 CFR 1.121(c)

This listing of claims will replace all prior versions, and listings, of claims in the application:

What is claimed is:

1. (Currently amended) In a database system employing a transaction log, an improved method for restoring databases to a consistent version, the method comprising:
providing a shared cache storing database blocks for use by multiple databases;
for a read-only transaction of a given database, creating a cache view of a given database ~~the shared cache~~ using the given database's transaction log, said cache view comprising particular database blocks of the shared cache that record a view of a particular version of the database at a given point in time;
creating a shadow cache for storing any database blocks that overflow said cache view during use of the cache view by the read-only transaction; ~~and~~
in conjunction with the cache view and the shadow cache, preserving a logical undo operation for the read-only transaction of the given database for logically undoing transactions which have begun but have yet to commit ~~, so as to allow the given database to be restored~~; ~~and~~
performing the logical undo operation in order to reconstruct to a transactionally consistent prior version of the given database upon starting the read-only transaction, thereby returning a result comprising a transactionally consistent version of the given database supporting read-only uses.
2. (Currently amended) The method of claim 1, wherein during occurrence of the read-only transaction any database blocks associated with the cache view are not written from the shared cache to the given database ~~back to disk~~.
3. (Original) The method of claim 1, wherein the shadow cache is implemented via a temporary database table.

4. (Canceled)
5. (Currently amended) The method of claim 1, wherein the shadow cache is used only in the event the cache view overflows the cache view~~if an overflow occurs~~.
6. (Currently amended) The method of claim 1, further comprising:
providing an allocation bitmap for indicating~~indicating~~ database blocks in use in the shadow cache.
7. (Currently amended) The method of claim 6, further comprising:
upon completion of the read-only transaction, deleting the shadow cache ~~simply~~ by updating the allocation bitmap for allocated database blocks ~~and then deleting the shadow table~~.
8. (Currently amended) The method of claim 1, wherein the shadow cache comprises a temporary database table including a first column for maintaining a block number of a cache view block having undo/redo records applied to it and a second column for maintaining a block number in a temporary database allocated to save off a modified block from the cache view~~saves off database blocks that are computationally expensive to recreate~~.
9. (Canceled)
10. (Original) The method of claim 1, further comprising:
upon termination of the read-only transaction, marking the cache view as closed.
11. (Currently amended) The method of claim 10, further comprising:
traversing the shared cache looking for database blocks to purge; and
reusing~~purging~~ database blocks from any cache view that has been marked as closed.

12. (Currently amended) The method of claim 1, further comprising:
reusing the cache view created for the read-only transaction for other read-only transactions, which start within a specified period of time following the start of the read-only transaction ~~if no new write operations have been committed~~.

13. (Currently amended) The method of claim 1, further comprising:
~~automatically~~ detecting the read-only transaction; and
upon occurrence of write operations, adding ~~posting~~ back link log records to the database's transaction log that serve to link together blocks of the transaction log that pertain to the read-only transaction.

14. (Original) The method of claim 13, further comprising:
if the read-only transaction must be undone, using the back link log records to skip portions of the transaction log that are irrelevant for undoing the read-only transaction, wherein the back link log records are only generated in the transaction log when there are active read only transactions.

15. (Original) A computer-readable medium having processor-executable instructions for performing the method of claim 1.

16. (Original) A downloadable set of processor-executable instructions for performing the method of claim 1.

17. (Currently amended) A database system capable of restoring databases to a consistent version, the system comprising:
a log manager module which manages a transaction log of the a-database system
~~employing a transaction log;~~
a cache manager module for managing a shared cache that stores database blocks for use by multiple databases and creating a cache view of a given database ~~the shared cache~~ created using the transaction log of thea given database, said cache view being created in response to a read-only transaction of thea given database, said cache view

comprising particular database blocks of the shared cache that record a view of a particular version of the database at a given point in time; wherein the cache manager utilizes a shadow cache for storing any database blocks that overflow said cache view during use of the cache view by the read-only transaction; and

a transaction manager module for performing read-only transactions of the database system and which performs ~~a module for preserving~~ a logical undo operation for the read-only transaction of the given database for logically undoing transactions which have begun but have yet to commit in order to reconstruct, so as to allow the given database to be restored to a transactionally consistent prior version of the given database upon starting the read-only transaction, thereby returning a result comprising a transactionally consistent version of the given database supporting read-only uses.

18. (Currently amended) The system of claim 17, wherein during occurrence of the read-only transaction any database blocks associated with the cache view are not written from the shared cache to the given database~~back to disk~~.

19. (Original) The system of claim 17, wherein the shadow cache is implemented via a temporary database table.

20. (Canceled)

21. (Currently amended) The system of claim 17, wherein the shadow cache is used only in the event the cache view overflows the cache view~~if an overflow occurs~~.

22. (Currently amended) The system of claim 17, wherein said cache manager maintains ~~further comprising~~ an allocation bitmap indicating database blocks in use in the shadow cache.

23. (Currently amended) The system of claim 22, wherein said cache manager ~~further comprising~~:

~~a module for deleting~~ deletes the shadow cache ~~simply~~ by updating the allocation

bitmap for allocated database blocks ~~and then deleting the shadow table.~~

24. (Currently amended) The system of claim 17, wherein the shadow cache comprises a temporary database table including a first column for maintaining a block number of a cache view block having undo/redo records applied to it and a second column for maintaining a block number in a temporary database allocated to save off a modified block from the cache viewsaves off database blocks that are computationally expensive to recreate.

25. (Canceled)

26. (Currently amended) The system of claim 17, wherein said cache manager marks further comprising:
~~a module for marking~~ the cache view as closed, upon termination of the read-only transaction.

27. (Currently amended) The system of claim 26, wherein said cache manager traverses further comprising:
~~a module for traversing~~ the shared cache looking for database blocks to purge, and purges database for reusing blocks from any cache view that has been marked as closed.

28. (Currently amended) The system of claim 17, wherein said cache manager reuses further comprising:
~~a module for reusing~~ the cache view created for the read-only transaction for other read-only transactions which start within a specified period of time following the start of the read-only transaction, ~~if no write operations have occurred.~~

29. (Currently amended) The system of claim 17, wherein said log manager detects further comprising:
~~a module for automatically detecting~~ the read-only transaction, and adds posting back link log records to the transaction log that serve to link together blocks of the

transaction log that pertain to the read-only transaction.

30. (Currently amended) The system of claim 29, wherein said log manager uses
~~further comprising:~~

~~a module for using~~ the back link log records to skip portions of the transaction log
that are irrelevant for undoing the read-only transaction ~~if the read-only transaction must~~
~~be undone~~, wherein the back link log records are only generated in the transaction log
when there are active read only transactions.